NAG C Library Function Document

nag_dggbal (f08whc)

1 Purpose

nag_dggbal (f08whc) balances a pair of real square matrices \( (A, B) \) of order \( n \). Balancing usually improves the accuracy of computed generalized eigenvalues and eigenvectors.

2 Specification

```c
void nag_dggbal (Nag_OrderType order, Nag_JobType job, Integer n, double a[], Integer pda, double b[], Integer pdb, Integer *ilo, Integer *ihi, double lscale[], double rscale[], NagError *fail)
```

3 Description

Balancing may reduce the 1-norms of the matrices, and improve the accuracy of the computed eigenvalues and eigenvectors in the real generalized eigenvalue problem

\[ Ax = \lambda Bx. \]

nag_dggbal (f08whc) is usually the first step in the solution of the above generalized eigenvalue problem. Balancing is optional but it is highly recommended.

The term ‘balancing’ covers two steps, each of which involves similarity transformations on \( A \) and \( B \). The function can perform either or both of these steps. Both steps are optional.

1. The function first attempts to permute \( A \) and \( B \) to block upper triangular form by a similarity transformation:

\[
P A P^T = F = \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{22} & F_{23} \\ F_{33} \end{pmatrix}
\]

\[
P B P^T = G = \begin{pmatrix} G_{11} & G_{12} & G_{13} \\ G_{22} & G_{23} \\ G_{33} \end{pmatrix}
\]

where \( P \) is a permutation matrix, \( F_{11}, F_{33}, G_{11}, \) and \( G_{33} \) are upper triangular. Then the diagonal elements of the matrix pairs \( (F_{11}, G_{11}) \) and \( (F_{33}, G_{33}) \) are generalized eigenvalues of \( (A, B) \). The rest of the generalized eigenvalues are given by the matrix pair \( (F_{22}, G_{22}) \) which are in rows and columns \( i_{lo} \) to \( i_{hi} \). Subsequent operations to compute the generalized eigenvalues of \( (A, B) \) need only be applied to the matrix pair \( (F_{22}, G_{22}) \); this can save a significant amount of work if \( i_{lo} > 1 \) and \( i_{hi} < n \). If no suitable permutation exists (as is often the case), the function sets \( i_{lo} = 1 \) and \( i_{hi} = n \).

2. The function applies a diagonal similarity transformation to \( (F, G) \), to make the rows and columns of \( (F_{22}, G_{22}) \) as close in norm as possible:

\[
D F D = \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{22} & F_{23} \\ F_{33} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix}
\]

\[
D G D = \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix} \begin{pmatrix} G_{11} & G_{12} & G_{13} \\ G_{22} & G_{23} \\ G_{33} \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & I \end{pmatrix}
\]

This transformation usually improves the accuracy of computed generalized eigenvalues and eigenvectors.

[NP3645/7] f08whc.1
4 References


5 Parameters

1:  \textbf{order} – Nag_OrderType  
\textit{Input}

\textit{On entry:} the \textbf{order} parameter specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by \textbf{order} = Nag_RowMajor. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this parameter.

\textit{Constraint:} \textbf{order} = Nag_RowMajor or Nag_ColMajor.

2:  \textbf{job} – Nag_JobType  
\textit{Input}

\textit{On entry:} specifies the operations to be performed on matrices \textit{A} and \textit{B}:

- \textbf{job} = Nag_DoNothing, no balancing is done. Initialize \textit{ilo} = 1, \textit{ihi} = \textit{n}, \text{lscale}_i = 1.0 and \text{rscale}_i = 1.0, for \textit{i} = 1, \ldots, \textit{n};
- \textbf{job} = Nag_Permute, only permutations are used in balancing;
- \textbf{job} = Nag_Scale, only scalings are are used in balancing;
- \textbf{job} = Nag_DoBoth, both permutations and scalings are used in balancing.

\textit{Constraint:} \textbf{job} = Nag_DoNothing, Nag_Permute, Nag_Scale or Nag_DoBoth.

3:  \textbf{n} – Integer  
\textit{Input}

\textit{On entry:} \textit{n}, the order of the matrices \textit{A} and \textit{B}.

\textit{Constraint:} \textbf{n} \geq 0.

4:  \textbf{a} – double  
\textit{Input/Output}

\textit{Note:} the dimension, \textit{dim}, of the array \textbf{a} must be at least max(1, \textit{pda} \times \textbf{n}).

Where \textit{A}(\textit{i}, \textit{j}) appears in this document, it refers to the array element

- \textbf{order} = Nag_ColMajor, \textbf{a}[(\textit{j} - 1) \times \textit{pda} + \textit{i} - 1];
- \textbf{order} = Nag_RowMajor, \textbf{a}[(\textit{i} - 1) \times \textit{pda} + \textit{j} - 1].

\textit{On entry:} the \textit{n} by \textit{n} matrix \textit{A}.

\textit{On exit:} \textbf{a} is overwritten by the balanced matrix.

\textbf{a} is not referenced if \textbf{job} = Nag_DoNothing.

5:  \textbf{pda} – Integer  
\textit{Input}

\textit{On entry:} the stride separating matrix row or column elements (depending on the value of \textbf{order}) in the array \textbf{a}.

\textit{Constraint:} \textbf{pda} \geq \text{max}(1, \textbf{n}).

6:  \textbf{b} – double  
\textit{Input/Output}

\textit{Note:} the dimension, \textit{dim}, of the array \textbf{b} must be at least max(1, \textit{pdb} \times \textbf{n}).

Where \textit{B}(\textit{i}, \textit{j}) appears in this document, it refers to the array element

- \textbf{order} = Nag_ColMajor, \textbf{b}[(\textit{j} - 1) \times \textit{pdb} + \textit{i} - 1];
- \textbf{order} = Nag_RowMajor, \textbf{b}[(\textit{i} - 1) \times \textit{pdb} + \textit{j} - 1].

\textit{On entry:} the \textit{n} by \textit{n} matrix \textit{B}.
On exit: b is overwritten by the balanced matrix. b is not referenced if job = Nag_DoNothing.

7: pdb – Integer

On entry: the stride separating matrix row or column elements (depending on the value of order) in the array b.

Constraint: pdb ≥ max(1, n).

8: ilo – Integer *

9: ihi – Integer *

On exit: ilo and ihi are set such that A(i, j) = 0 and B(i, j) = 0 if i > j and 1 ≤ j < ilo or ihi < i ≤ n.

If job = Nag_DoNothing or Nag_Scale, ilo = 1 and ihi = n.

10: lscale[dim] – double

Note: the dimension, dim, of the array lscale must be at least max(1, n).

On exit: details of the permutations and scaling factors applied to the left side of the matrices A and B. If Pi is the index of the row interchanged with row i and di is the scaling factor applied to row i, then

\[ lscale[i - 1] = P_i, \text{ for } i = 1, \ldots, \text{ilo} - 1; \]
\[ lscale[i - 1] = d_i, \text{ for } i = \text{ilo}, \ldots, \text{ihi}; \]
\[ lscale[i - 1] = P_i, \text{ for } i = \text{ihi} + 1, \ldots, n. \]

The order in which the interchanges are made is n to ihi + 1, then 1 to ilo - 1.

11: rscale[dim] – double

Note: the dimension, dim, of the array rscale must be at least max(1, n).

On exit: details of the permutations and scaling factors applied to the right side of the matrices A and B. If Pj is the index of the column interchanged with column j and dj is the scaling factor applied to column j, then

\[ rscale[j - 1] = P_j, \text{ for } j = 1, \ldots, \text{ilo} - 1; \]
\[ rscale[j - 1] = d_j, \text{ for } j = \text{ilo}, \ldots, \text{ihi}; \]
\[ rscale[j - 1] = P_j, \text{ for } j = \text{ihi} + 1, \ldots, n. \]

The order in which the interchanges are made is n to ihi + 1, then 1 to ilo - 1.

12: fail – NagError *

The NAG error parameter (see the Essential Introduction).

6 Error Indicators and Warnings

NE_INT

On entry, n = \langle value\rangle.

Constraint: n ≥ 0.

On entry, pda = \langle value\rangle.

Constraint: pda > 0.

On entry, pdb = \langle value\rangle.

Constraint: pdb > 0.
NE_INT_2
On entry, pda = (value), n = (value).
Constraint: pda ≥ max(1, n).
On entry, pdb = (value), n = (value).
Constraint: pdb ≥ max(1, n).

NE_ALLOC_FAIL
Memory allocation failed.

NE_BAD_PARAM
On entry, parameter (value) had an illegal value.

NE_INTERNAL_ERROR
An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

7 Accuracy
The errors are negligible, compared to those in subsequent computations.

8 Further Comments
nag_dggbal (f08whc) is usually the first step in computing the real generalized eigenvalue problem but it is an optional step. The matrix B is reduced to the upper triangular form using the QR factorization function nag_dgeqrf (f08aec) and this orthogonal transformation Q is applied to the matrix A by calling nag_dormqr (f08agc). This is followed by nag_dgghrd (f08wec) which reduces the matrix pair into the generalized Hessenberg form.

If the matrix pair (A, B) is balanced by this function, then any generalized eigenvectors computed subsequently are eigenvectors of the balanced matrix pair. In that case, to compute the generalized eigenvectors of the original matrix, nag_dgggbak (f08wjc) must be called.

The total number of floating-point operations is approximately proportional to n^2.
The complex analogue of this function is nag_zggbal (f08wvc).

9 Example
See Section 9 of the documents for nag_dhgeqz (f08xec) and nag_dtgevc (f08ykc).